

Heute geht's um...

# Krawatten



<http://commons.wikimedia.org/wiki/Image:Necktie.jpg>



...äähh...

Tie's

## Tie's sind...

- Eine „gebundene“ Variable
- Erzeugt „Magie“
- Session-Objekt bei Apache::Session ist ein „tied“ Hash
  - Eigentliches „tie“ wird aber „missbraucht“
- Hat mir bei Datei-Änderungen geholfen

## Ein häufiges Beispiel

```
#!/usr/bin/perl
```

```
use strict;
```

```
use warnings;
```



```
use Tie::File;
```

```
my $file = '/path/to/file';
```



```
tie my @lines, 'Tie::File', $file;
```

```
print $_ for @lines;
```



```
untie @lines;
```

## Die Session...

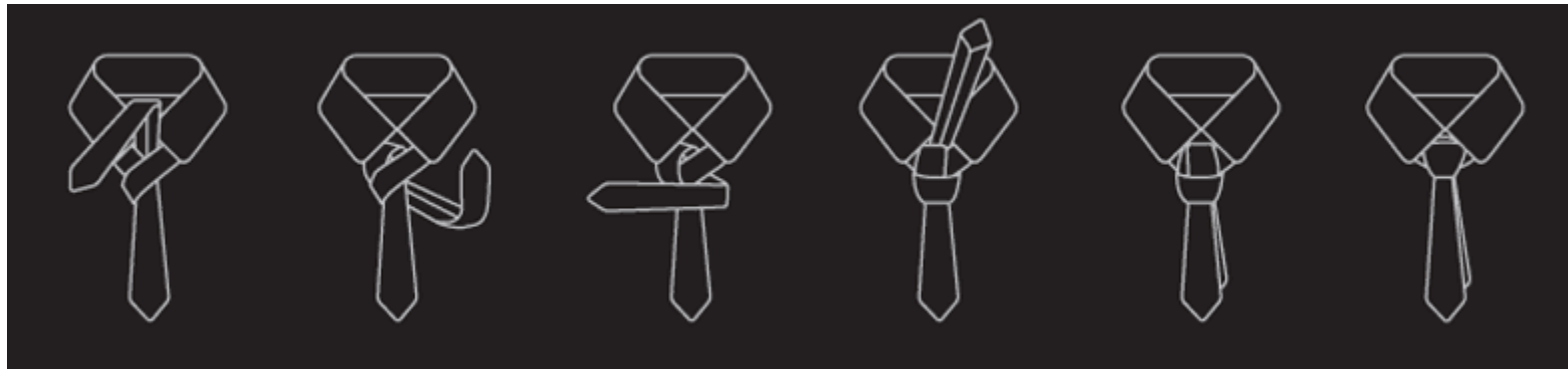
```
# Kein DSMS. Also Session in der DB.
tie %session, $connector, $id, {
    Handle => $dbh,
    Commit => 1,
    LongReadLen => 512*1024
};

#...

tied(%{$self})->delete;
```

## Die Schritte zum Tie...

Ein Hash, der Zugriffe in eine Datei loggt...



# Schritt 1

## Package definieren

```
package Tie::Eecture
```

## Initialisieren des Hashs

```
sub TIEHASH{  
    my ($class,$file) = @_;  
    my $self = bless {}, $class;  
    $self->logger( $file );  
    return $self;  
}
```



## Im Programm:

```
tie my %hash, 'Tie::Eecture', 'ein/Datei.name');
```

## Schritt 2

Wie sollen Werte gespeichert werden?

```
sub STORE{  
    my ($self,$key,$value) = @_;  
    $self->{hash}->{$key} = $value;  
}
```

Im Programm:

```
$hash{Schluessel} = 'Wert';
```



## Schritt 3

Wie sollen Werte ausgelesen werden?  
Hier kommt auch das Logging hin...

```
sub FETCH{  
    my ($self,$key) = @_;  
  
    $self->log( $key );  
    return $self->{hash}->{$key};  
}
```

Im Programm:

```
print $hash{Schluessel};
```



## Schritt 4

Wie sollen Werte gelöscht werden?

```
sub DELETE{  
    my ($self,$key) = @_;  
    delete $self->{hash}->{$key};  
}
```

Im Programm:

```
delete $hash{Schluessel};
```



## Schritt 5

Weitere Methoden implementieren und ständig  
Verbessern...

... Live-Demo



## Das Modul...

```
package Tie::Beispiel;

sub TIEHASH{
    my ($class,$file) = @_;
    my $self = bless {}, $class;
    $self->logger( $file );
    return $self;
}

sub STORE{
    my ($self,$key,$value) = @_;
    $self->{hash}->{$key} = $value;
}

sub FETCH{
    my ($self,$key) = @_;
    $self->log( $key );
    return $self->{hash}->{$key};
}

sub DELETE{
    my ($self,$key) = @_;
    delete $self->{hash}->{$key};
}

sub logger{ # open filehandle and store it }
sub log{ # print into log file }
```

## Aber Achtung!!

- Werte, die vorher in der Variablen waren werden gelöscht
- ... Und nicht wieder hergestellt.
- Man kann noch viel mehr machen...
  - EXISTS
  - FIRSTKEY
  - NEXTKEY
  - UNTIE
  - ...

## Wann lohnt sich ein tie??

- wenn eine Variable ein „magisches“ Verhalten braucht...
- Wenn ein Objekt zu viel wäre...

## Tie-Module bei NF3-Umstellung

- Tie::File
  - Automatische Konvertierung von Pfaden
  - Einfügen von globalen Variablen
  - Einfügen von „use strict“

# Tie-Module in Anwendung

```
sub search_replace{
  my @files = @_;

  my $counter = 0;
  for my $file ( @files ){
    next unless $file =~ /\.(cgi|pm|pl|html)$/;
    print $file, "\n";
    sleep 1;
    tie my @lines, 'Tie::File', $file or next;
    unless( grep{ $_ =~ /use strict/ }@lines ){
      $lines[0] =~ s~(#!/usr/bin/perl -T)~$1\n\nuse strict;~;
    }
    for my $line ( @lines ){
      $line =~ s!/shared/img!/shared/images!/g;
      $line =~ s!/shared/css!/shared/stylesheets!/g;
      $line =~ s!my \%(felder) = setFelder!my \%$1;\n\%$1 = setFelder!g;
      $line =~ s!(require "..\settings.pl";)!our_variables().$1!e;
    }
    untie @lines;
  }
}
```

## Weitere Ties

- Man kann auch andere Variablen „tie“
  - Arrays
  - Skalare
  - Filehandles
- Es gibt viele Tie::\*-Module
  - Tie::File
  - Tie::DBI
  - Tie::IxHash
  - ...
- <http://perldoc.perl.org/functions/tie.html>
- <http://perldoc.perl.org/perltie.html>